



SKILLFACTORY

Тестирование

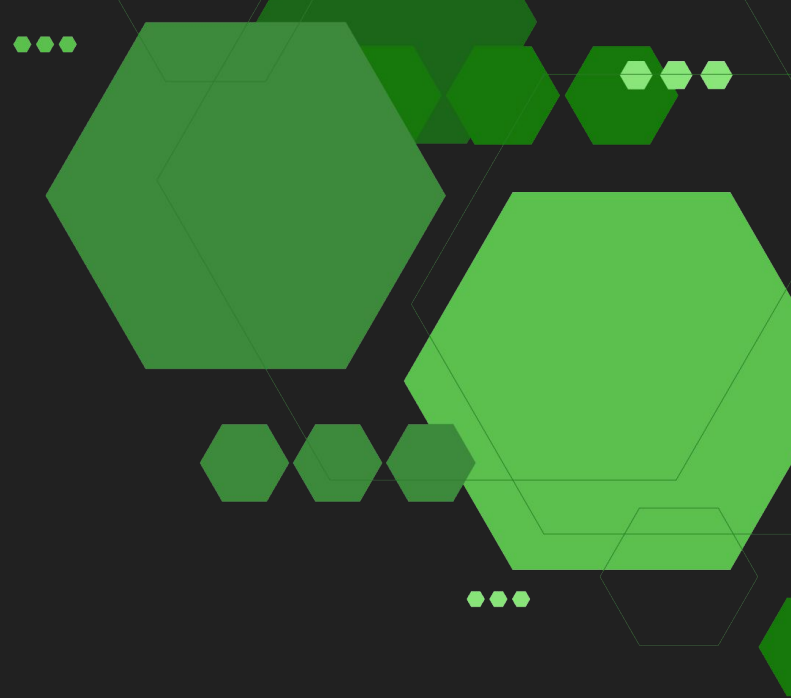
Николай Троицкий
Эксперт по Data Engineering

Николай Троицкий

- С 2005 года работаю в ИТ сфере
- Занимался функциональным тестированием на проектах внедрения продуктов Oracle
- Работал дата-инженером в Леруа Мерлен Россия.
- И занимаюсь ИТ-консалтингом и маркетинговыми исследованиями в собственной компании IT Boutique



SKILLFACTORY



01. Введение

Цена ошибки: больше чем финансовые потери



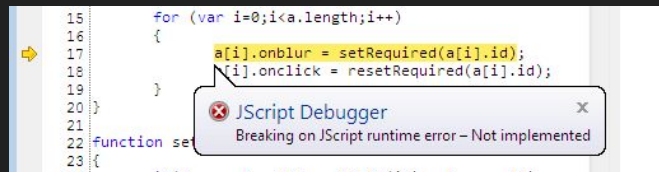
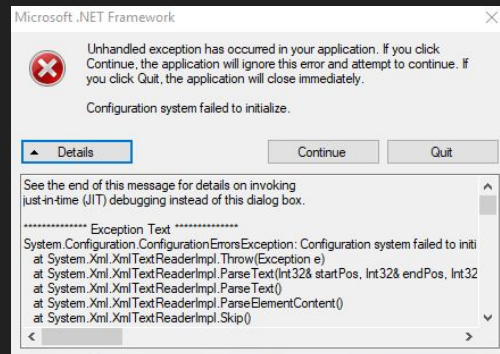
О чем эта лекция?

- Что и зачем мы тестируем?
- Какие бывают виды тестирования?
- Как ошибки и баги проявляют себя?
- Ошибка (error) и исключительная ситуация (exception). В чем различия? Как мы контролируем нештатные ситуации в коде?
- Какие у нас есть инструменты, чтобы находить и исправлять ошибки?

Смысл тестирования

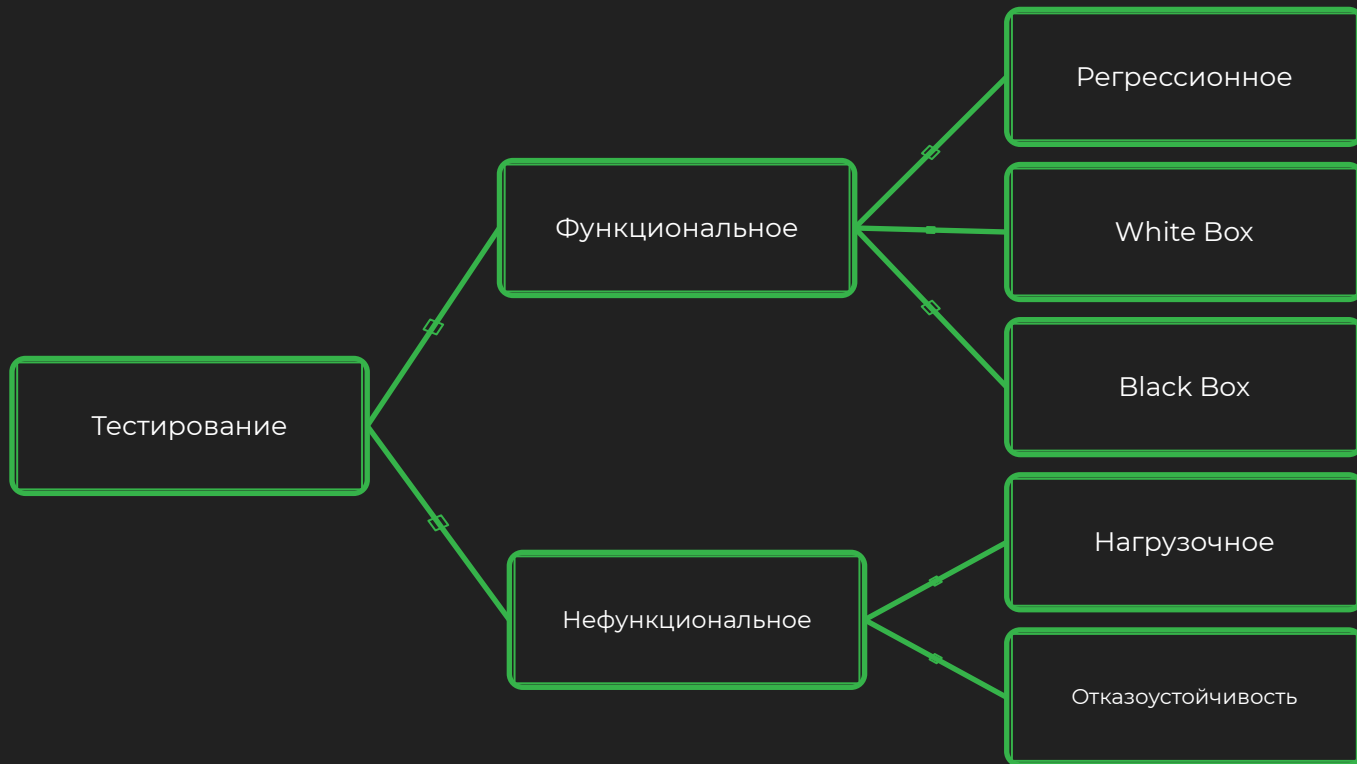
Проверить выполнение требований к программному коду:

- Он работает – мы можем запустить нашу программу и она отработает, а не «упадет» с непредсказуемой ошибкой
- Он имеет все необходимые функции – реализовано все, что требовалось.
- Наши вычисления ВСЕГДА дают правильный результат – мы можем воспроизвести расчет и убедиться, что значения совпадут.
- Наш код отработывает за требуемое время



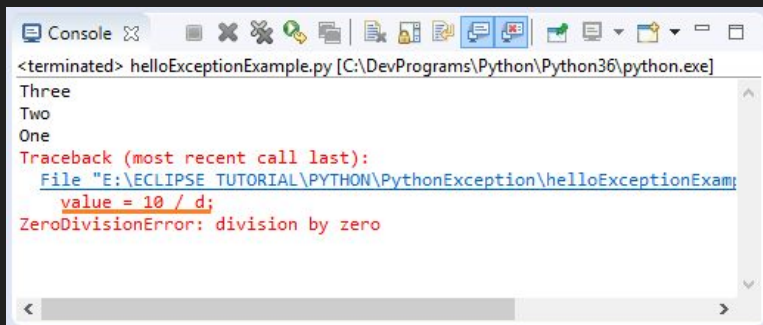
$$8 \div 2(2 + 2) = ?$$

Классификация тестирования



02. Основные подходы к тестированию

Чтение сообщений об ошибках



```
<terminated> helloExceptionExample.py [C:\DevPrograms\Python\Python36\python.exe]
Three
Two
One
Traceback (most recent call last):
  File "E:\ECLIPSE TUTORIAL\PYTHON\PythonException\helloExceptionExample.py", line 10, in <module>
    value = 10 / d;
ZeroDivisionError: division by zero
```

Runtime Error:

Traceback (most recent call last):

File "", line 2, in
print(mylist[10])

IndexError: list index out of range

```
C:\pilfer-archive>python pilfer-archive-new.py
[*] Found 9162340 entries for pdf
Traceback (most recent call last):
  File "pilfer-archive-new.py", line 151, in <module>
    main()
  File "pilfer-archive-new.py", line 146, in main
    t1 = Thread(target=Queues().search_worker(),name="search-%d" % (i)).start()
  File "pilfer-archive-new.py", line 114, in search_worker
    titles = Discover().get_titles(item,numFound)
  File "pilfer-archive-new.py", line 77, in get_titles
    data = json.loads(urllib2.urlopen("https://archive.org/advancedsearch.php?q=%s&mediatype=&rows=%d&page=1&output=json&save=no#raw" % (term,numFound)).read())

  File "C:\Python27\Lib\socket.py", line 358, in read
    buf.write(data)
MemoryError: out of memory
```

СТЭК ВЫЗОВОВ

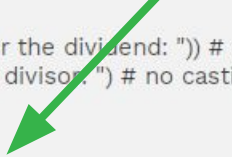
```
Enter the dividend: 5
Enter the divisor: 3
Traceback (most recent call last):
  File "typeError.py", line 12, in <module>
    result = compute_division()
  File "typeError.py", line 6, in compute_division
    result = dividend/divisor
TypeError: unsupported operand type(s) for /: 'int' and 'str'
```

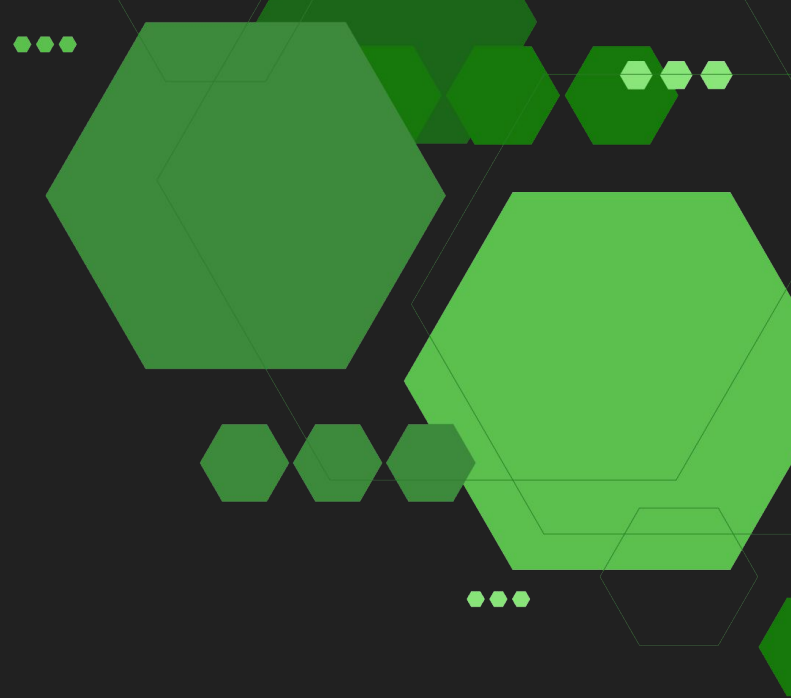
```
def compute_division():
    dividend = int(input("Enter the dividend: ")) # cast string to int
    divisor = input("Enter the divisor: ") # no casting

    # Compute division
    result = dividend/divisor

    # print result
    print("The result of {}/{} is: {}".format(dividend, divisor, result))

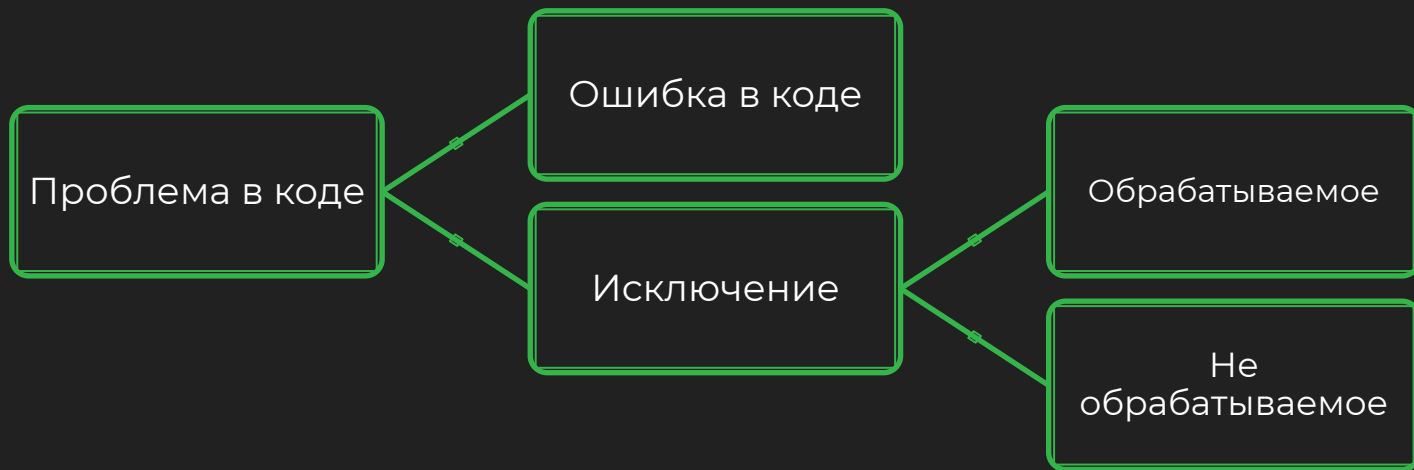
if __name__ == '__main__':
    result = compute_division()
```



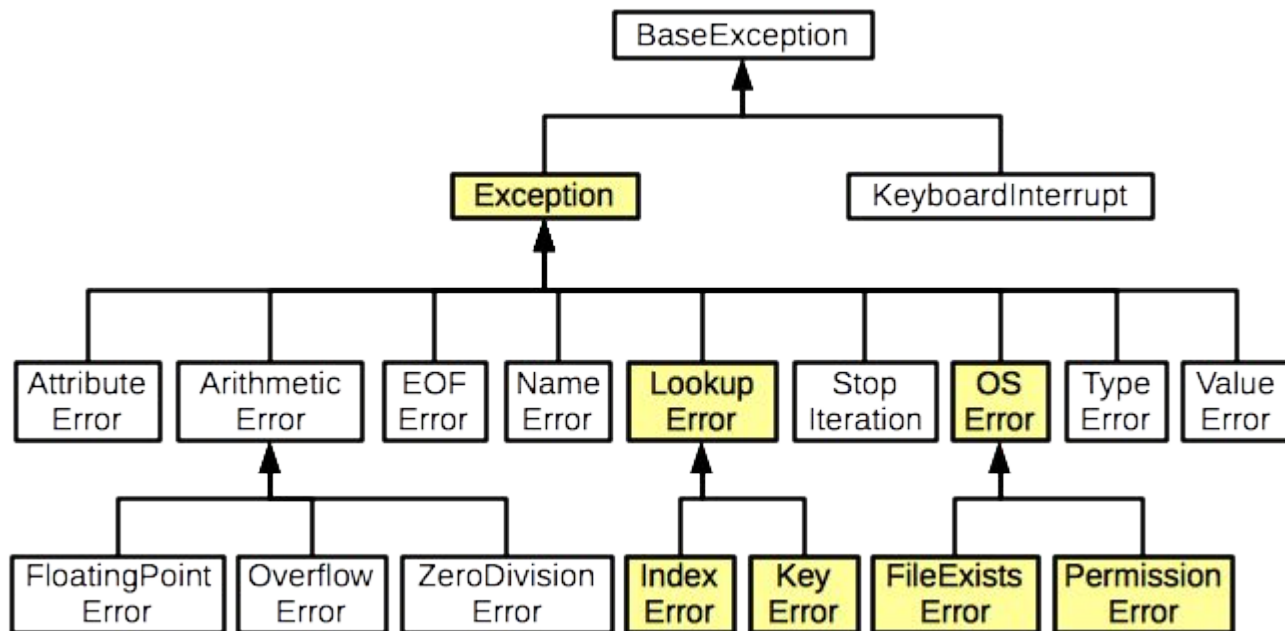


03. Что такое исключение?

Что такое исключение?



Иерархия исключений



Использование исключений

```
1 def divide(x, y):
2     try:
3         print(f'{x}/{y} is {x / y}')
4     except ZeroDivisionError as e:
5         print(e)
6     else:
7         print("divide() function worked fine.")
8
9
10 divide(10, 2)
11 divide(10, 0)
12 divide(10, 4)
13
```

Run: exception_handling

```
/Users/pankaj/Documents/PycharmProjects/PythonTutorialPro/
10/2 is 5.0
divide() function worked fine.
division by zero
10/4 is 2.5
divide() function worked fine.

Process finished with exit code 0
```

Детали TRY...EXCEPT

{

Run this code

{

Execute this code when
there is an exception

{

No exceptions? Run this
code.

{

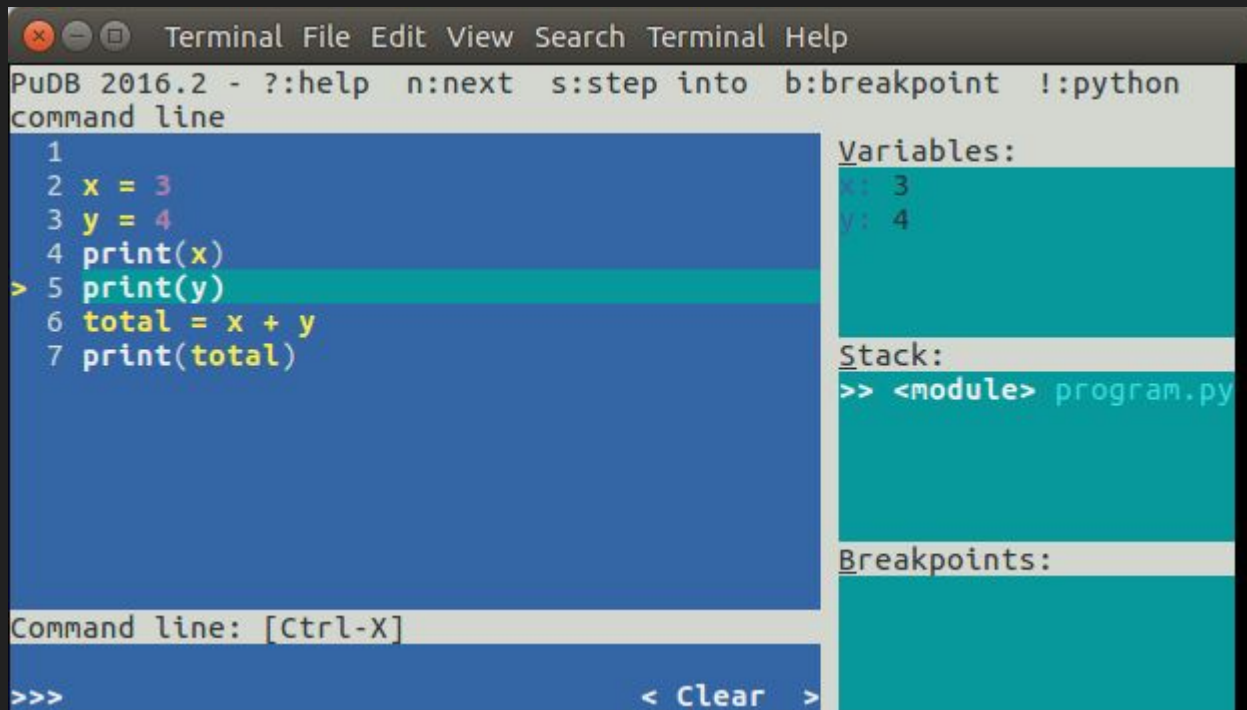
Always run this code.

```
>>> f = open('1.txt')
>>> ints = []
>>> try:
...     for line in f:
...         ints.append(int(line))
... except ValueError:
...     print('Это не число. Выходим.')
... except Exception:
...     print('Это что ещё такое?')
... else:
...     print('Всё хорошо.')
... finally:
...     f.close()
...     print('Я закрыл файл.')
...     # Именно в таком порядке: try, группа except, затем
...     else, и только потом finally.
...
Это не число. Выходим.

Я закрыл файл.
```

04. Введение в отладку кода

Отладка с помощью print



```
Terminal File Edit View Search Terminal Help
PuDB 2016.2 - ?:help  n:next  s:step into  b:breakpoint  !:python
command line
1
2 x = 3
3 y = 4
4 print(x)
> 5 print(y)
6 total = x + y
7 print(total)

Variables:
x: 3
y: 4

Stack:
>> <module> program.py

Breakpoints:

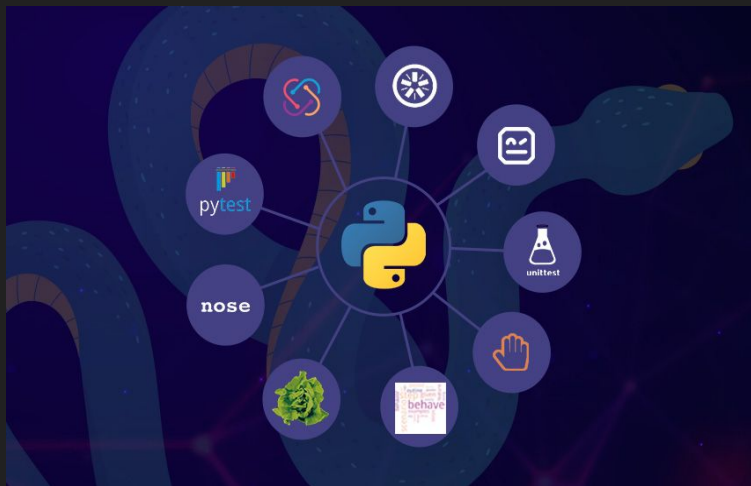
Command line: [Ctrl-X]

>>> < Clear >
```

Отладка с помощью pdb

```
$ python -m pdb example.py
> example.py(1)<module>()
-> my_string = 'one'
(Pdb) next
1 -> my_string = 'one'
2     my_other_string = 2
3
4     print my_string + my_other_string
> example.py(2)<module>()
-> my_other_string = 2
(Pdb) next
(Pdb) next
1     my_string = 'one'
2     my_other_string = 2
3
4 -> print my_string + my_other_string
TypeError: "cannot concatenate 'str' and 'int' objects"
```

Фреймворки для тестирования кода Python



```
$ pytest
===== test session starts =====
platform linux -- Python 3.x.y, pytest-6.x.y, py-1.x.y, pluggy-0.x.y
cachedir: $PYTHON_PREFIX/.pytest_cache
rootdir: $REGENDOC_TMPDIR
collected 1 item

test_sample.py F [100%]

===== FAILURES =====
_____ test_answer _____

    def test_answer():
>     assert inc(3) == 5
E       assert 4 == 5
E       + where 4 = inc(3)

test_sample.py:6: AssertionError
===== short test summary info =====
FAILED test_sample.py::test_answer - assert 4 == 5
===== 1 failed in 0.12s =====
```